3-я Всероссийская ежегодная конференция

«Юные техники и изобретатели»



**Проект «Электронный органайзер»**

***Информационные технологии***

**выполнил:**

ученик 5 класса №3

Брянского городского лицея №1

имени А.С. Пушкина

Лукьянович Роберт

**руководители:**

Корникова Т.М., учитель информатики

Киреев А.М., руководитель IT-парк

**Брянск 2016**

Оглавление

[Введение 3](#_Toc446260571)

[Глава 1. Электронные органайзеры 4](#_Toc446260572)

[Глава 2. Описание проекта 7](#_Toc446260573)

[1.1. Среда программирования 7](#_Toc446260574)

[1.2. Интерфейс приложения 8](#_Toc446260575)

[1.3. Программный код 9](#_Toc446260576)

[1.4. Возможности приложения и практическое применение 18](#_Toc446260577)

[Плюсы электронного ежедневника 1](#_Toc446260579)9

[Минусы электронного ежедневника 19](#_Toc446260580)

[Заключение 20](#_Toc446260578)

[Список Интернет-ресурсов:](#_Toc446260581) 21

# Введение

Я, ребёнок 21 века, не могу и дня прожить без компьютера и мобильного телефона.Мне интересно всё, что связано с IT-технологиями, но больше всего я хочу создавать собственные программы и приложения. Поэтому ещё в прошлом году я стал изучать язык программирования С#.

Поступив в 5 класс Брянского городского лицея №1 им. А.С. Пушкина, я с головой окунулся в лицейскую жизнь, и понял, что могу утонуть в потоке событий и информации, простого ученического дневника не хватает. В школьный дневник изменения расписания, как правило, не вносят: это портит «лицо ученика». А в голове не всё удержишь. Так происходит некоторая путаница, в том числе и с домашним заданием. Сегодня ребята уже не пользуются блокнотами, поскольку на замену пришли планшеты и смартфоны. Так, согласно опросу 90 лицеистов, только 13 записывают домашнее задание в дневник или тетрадь. Остальные фотографируют на планшет или смартфон.

Но, кроме фотографирования расписания или домашнего задания, в арсенале таких гаджетов больше ничего нет. Занимаясь программированием, я придумал решение этой задачи.

Ответственному школьнику, лицеисту необходим органайзер для упорядочения, и не просто записная книжка, а электронное приложение. Поискав в Интернете, решил сам создать собственную модель органайзера, и адаптировать её для особенностей учебного процесса в лицее.

**Цель** **работы:** создание интерактивного приложения «Электронный органайзер» для учащихся учебных заведений.

**Основные этапы работы над проектом:**

1. Изучение различных видов электронных дневников и органайзеров.
2. Изучение теоретических основ создания приложений на языке С# в среде MS Visual Studio 2015.
3. Знакомство с основными компонентами, их свойствами и методами объектно-ориентированной среды языка С#.
4. Разработка интерфейса приложения «Электронный органайзер».
5. Написание программного кода.
6. Тестирование приложения «Электронный органайзер».
7. Изучение расширения возможностей приложения «Электронный органайзер».

# Глава 1. Электронные органайзеры

На современном этапе развития компьютерных технологий многие (если не большинство) пользователей всё чаще отказываются от ведения различного рода заметок, запланированных задач, встреч, напоминаний, паролей или кодов доступа к банковским картам на бумаге. Понятное дело, что удержать в голове такое огромное количество информации просто невозможно. Да и бумажные носители в последнее время стали весьма ненадёжны, поскольку, не исключается вариант утери или обычного случайного выбрасывания в мусорное ведро. И многие разработчики программного обеспечения решили компенсировать все эти неудобства и начали создавать электронные органайзеры. Сейчас в сети Интернет можно найти множество программ такого типа и скачать их бесплатно. При этом бесплатные продукты не на много отличаются от платных аналогов. Принципы, использованные для хранения необходимой информации, практически у всех программ одинаковы. Однако приложения такого типа принципиально делятся на две категории. К первой можно отнести стационарные и портативные версии, которые работают на компьютерах или мобильных устройствах.

Ко второй группе относятся программы стационарных и портативных версий, которые для хранения данных используют удаленные серверы. В первой группе программа устанавливается на компьютер или работает со съемного носителя. Обычно, интерфейс таких приложений напоминает бумажные блокноты. Разница может заключаться только в оформлении и каких-либо дополнительных возможностях. Многие категории данных разделены по группам. Это могут быть встречи, список дел по дням или по часам, дни рождений, календарь, хранилище персональных кодов и паролей и многое другое. Здесь отличие программ состоит только в фантазии разработчиков. Объединяющим фактором таких приложений является, в большинстве случаев, наличие напоминаний (особенно это касается мобильных устройств). Именно поэтому, пользователь никогда не пропустит важное событие. Однако есть и недостаток: если программа работает на компьютере, получить доступ можно только с этого компьютера или посредством локальной сети.

Вторая группа программ, напротив, предоставляет полный доступ к данным с любого компьютера, подключенного к сети Интернет. Эти программы используют для хранения информации удалённые сервера. После регистрации пользователю выделяется некоторое дисковое пространство, где, собственно, и хранится вся информация. Причем, это могут быть не только заметки, но и графические, и файлы мультимедиа. Также, предусмотрена синхронизация с мобильными устройствами. Думается, этот вариант является более удобным, поскольку доступ к данным может осуществляться из любой точки мира. Бесплатные программы такого типа довольно сильно распространены в Интернете, так что, скачать их проблемой не является. Второй вариант предпочтительнее. Здесь единственным ограничением является только объём выделяемого дискового пространства на удалённом сервере.

|  |  |
| --- | --- |
| http://www.linuxshop.ru/upload/iblock/f6b/f6ba774e1ef487e3b5f782f257f5e9bd.png | **LeaderTask** позволяет отобразить календарь на день, несколько дней, неделю, 2 недели. Причем всегда можно видеть как список задач, так и календарь, и в случае необходимости, переносить задачи из списка в календарь на нужное время. |
| EfficientPIM | **EfficientPIM** функциональный набор программы настолько широк, что теперь её можно назвать даже ежедневником или планировщиком задач, не только на определённый день, но и на некоторый, указанный срок наперёд. |
| TimeLeft | **TimeLeft** - программа, представляющая собой одновременно и системные часы, и органайзер с возможностью планирования событий и отслеживания времени, которое осталось до какой-либо даты, либо прошедшего после нее. По большому счёту, приложение более всего похоже на специфичные программные продукты, которые применяются в мобильных устройствах. |

Изучив предложенные варианты органайзеров, я понял, то ни один из них не подходит, так как мне нужно не только деление на дни, недели, месяцы, но и уроки. Мой органайзер должен быть похож на обычный школьный дневник, но с более широкими возможностями.

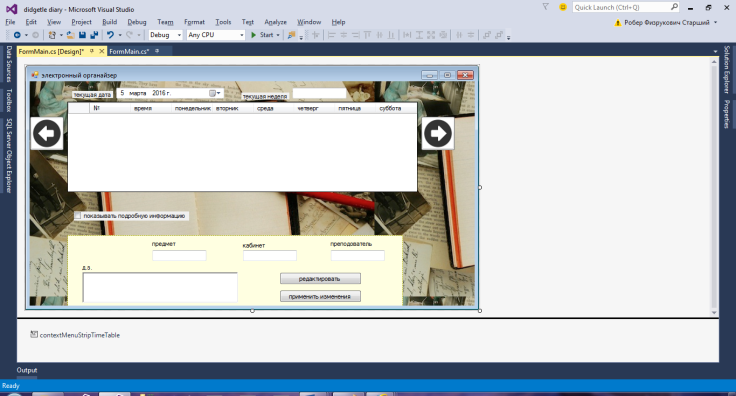
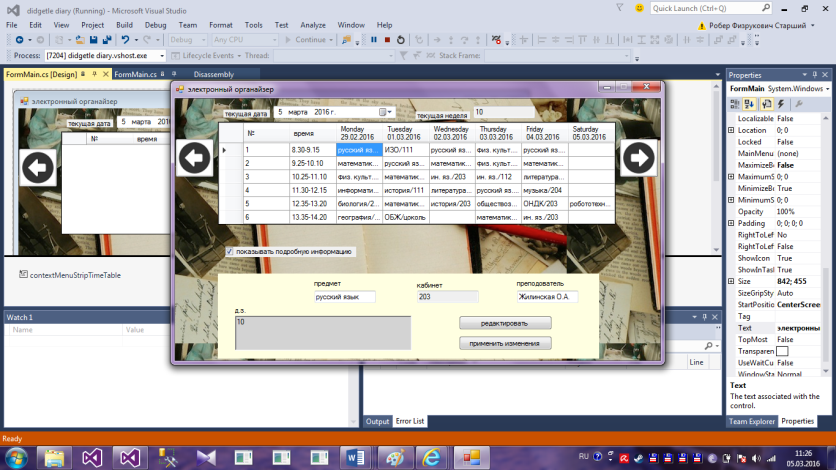
Преимущество моего органайзера, по сравнению с уже созданными, например, с **EfficientPIM**, в простоте внесения изменений, в более привычной для ученика структуре и понятном интерфейсе.

# Глава 2. Описание проекта

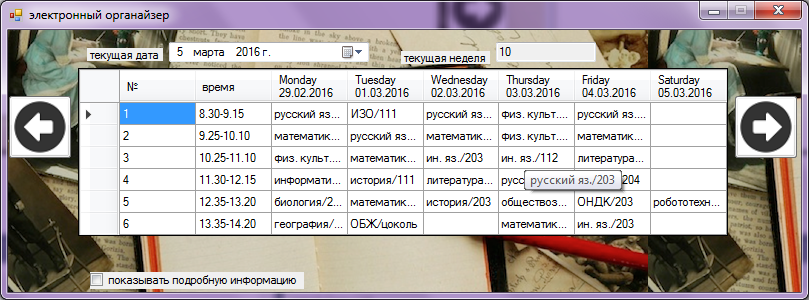
## Среда программирования

Для создания собственный органайзера я выбрал язык С# в среде Microsoft Visual Studio 2015. Эта среда предоставляет широкие возможности для сборки и развертывания приложения для Windows на любой платформе. Сначала я познакомился с основными возможностями С#. Изучил синтаксис языка, типы данных, классы, свойства, методы, атрибуты, основные этапы создания приложений: разработка алгоритма, конструирование интерфейса, написание программного кода. Создал множество простейших проектов: математический калькулятор, электронное табло для уроков и конкурсов, игры. Почти год работы, и я решил, что нужно попробовать свои силы в более серьезной работе.

## Интерфейс приложения

Основную часть формы занимает таблица 6×8, в которой 1 столбец это номер урока, 2 столбец отображает время урока, остальные 6 столбцов (дни недели с указанием даты) – сведения об уроках, кабинетах, учителях, домашнем задании заполняются пользователем. Для этого необходимо встать в соответствующую ячейку таблицы, выбрать «дополнить», в открывшемся расширении формы, заполнить поля «Предмет», «Кабинет», «Преподаватель», «Домашнее задание» и сохранить информацию, нажав кнопку «применить изменения». Для внесения изменений есть возможность редактирования информации.

В верхней части формы поле текущей даты, которое заполняется выбором даты в календаре. Поле «Текущая неделя» заполняется автоматически после установки даты.

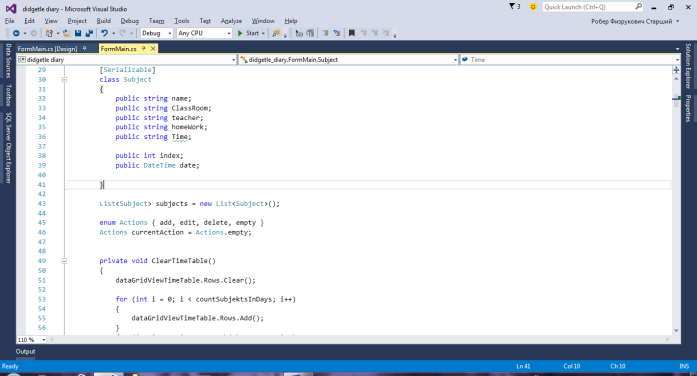


Для перехода к следующей или предыдущей неделе слева и справа от таблицы установлены стрелки перелистывания вперед и назад.

Компактный размер формы 842 × 455 позволяет приложение размещать в любой области экрана. При этом внесённая информация хорошо читается.

## 1.3. Программный код

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Runtime.Serialization.Formatters.Binary;

namespace didgetle\_diary

{

public partial class FormMain : Form

{

const int countDaysInWeek = 6;

const int countSubjektsInDays = 6;

const string bdFileName = "subject.bin";

DateTime[] days = new DateTime[countDaysInWeek];

int iIndex, JIndex;

[Serializable]

class Subject

{

public string name;

public string ClassRoom;

public string teacher;

public string homeWork;

public string Time;

public int index;

public DateTime date;

}

List<Subject> subjects = new List<Subject>();

enum Actions { add, edit, delete, empty }

Actions currentAction = Actions.empty;

private void ClearTimeTable()

{

dataGridViewTimeTable.Rows.Clear();

for (int i = 0; i < countSubjektsInDays; i++)

{

dataGridViewTimeTable.Rows.Add();

}

for (int i = 0; i < countSubjektsInDays; i++)

{

dataGridViewTimeTable.Rows[i].Cells[0].Value = i + 1;

dataGridViewTimeTable.Rows[0].Cells[1].Value = "8.30-9.15";

dataGridViewTimeTable.Rows[1].Cells[1].Value = "9.25-10.10";

dataGridViewTimeTable.Rows[2].Cells[1].Value = "10.25-11.10";

dataGridViewTimeTable.Rows[3].Cells[1].Value = "11.30-12.15";

dataGridViewTimeTable.Rows[4].Cells[1].Value = "12.35-13.20";

dataGridViewTimeTable.Rows[5].Cells[1].Value = "13.35-14.20";

}

}

public void DateToNumberWeek()

{

DateTime chooseDate = dateTimePickerCurrentDate.Value;

DateTime firstDayOfYear = new DateTime(chooseDate.Year, 1, 1);

int delta = 7 - Convert.ToInt32(firstDayOfYear.DayOfWeek);

int numberWeek = (chooseDate.DayOfYear + delta) / 7 + 1;

textBoxCurrentWeek.Text = numberWeek.ToString();

}

private void DateToWeek()

{

DateTime chooseDay = dateTimePickerCurrentDate.Value;

int numberDay = Convert.ToInt32(chooseDay.DayOfWeek);

DateTime startDay = chooseDay.AddDays(-numberDay + 1);

for (int i = 0; i < countDaysInWeek; i++)

{

days[i] = startDay;

startDay = startDay.AddDays(1);

}

}

private void FillHeaders()

{

for (int i = 0; i < countDaysInWeek; i++)

{

dataGridViewTimeTable.Columns[i + 2].HeaderText = String.Format("{0} {1}", days[i].DayOfWeek, days[i].ToShortDateString());

}

}

private void GoToNextWeek()

{

for (int i = 0; i < countDaysInWeek; i++)

{

days[i] = days[i].AddDays(7);

}

}

private void GoToPrevWeek()

{

for (int i = 0; i < countDaysInWeek; i++)

{

days[i] = days[i].AddDays(-7);

}

}

private void OnEdite()

{

textBoxClassRomm.ReadOnly =false;

textBoxCurrentWeek.ReadOnly = false;

textBoxCurrentWeek.ReadOnly = false;

richTextBoxHomeWork.ReadOnly = false;

richTextBoxHomeWork.BackColor = Color.White;//костыль

}

private void offEdit()

{

textBoxClassRomm.ReadOnly = true;

textBoxCurrentWeek.ReadOnly = true;

textBoxCurrentWeek.ReadOnly = true;

richTextBoxHomeWork.ReadOnly = true;

richTextBoxHomeWork.BackColor = Color.LightGray;

}

private void SevesubjectsToFile()

{

FileStream fs = new FileStream(bdFileName, FileMode.OpenOrCreate);

BinaryFormatter bf = new BinaryFormatter();

bf.Serialize(fs, subjects);

fs.Close();

}

private void LoudSubjectsFormFile()

{

FileInfo fi = new FileInfo(bdFileName);

if (fi.Exists==true)

{

FileStream fs = new FileStream(bdFileName, FileMode.Open);

BinaryFormatter bf = new BinaryFormatter();

subjects =(List<Subject>) bf.Deserialize(fs);

fs.Close();

}

}

public FormMain()

{

InitializeComponent();

}

private void checkBoxShowExInformation\_CheckedChanged(object sender, EventArgs e)

{

if (checkBoxShowExInformation.Checked == true)

{

panelAddEdit.Show();

this.CenterToScreen();

}

if (checkBoxShowExInformation.Checked == false)

{

panelAddEdit.Hide();

this.CenterToScreen();

}

}

private void PrintSubjects()

{

for (int i = 0; i < countDaysInWeek; i++)

{

for (int j = 0; j < subjects.Count; j++)

{

if (days[i].Year== subjects[j].date.Year&&

days[i].Month == subjects[j].date.Month&&

days[i].Day == subjects[j].date.Day)

{

dataGridViewTimeTable.Rows[subjects[j].index].Cells[i + 2].Value = subjects[j].name + "/" + subjects[j].ClassRoom;

}

}

}

}

private int GetSubjectIndex()

{

for( int i = 0; i < subjects.Count;i++)

{

if (subjects[i].date.Year==days[JIndex-2].Year&&

subjects[i].date.Month== days[JIndex - 2].Month&&

subjects[i].date.Day == days[JIndex - 2].Day)

{

return i;

}

}

return -1;

}

private void FormMain\_Load(object sender, EventArgs e)

{

ClearTimeTable();

dateTimePickerCurrentDate.Value = DateTime.Now;

DateToWeek();

FillHeaders();

DateToNumberWeek();

offEdit();

LoudSubjectsFormFile();

PrintSubjects();

}

private void buttonRight\_Click(object sender, EventArgs e)

{

ClearTimeTable();

dateTimePickerCurrentDate.Value = dateTimePickerCurrentDate.Value.AddDays(7);

DateToNumberWeek();

DateToWeek();

FillHeaders();

PrintSubjects();

}

private void buttonLeft\_Click(object sender, EventArgs e)

{

ClearTimeTable();

dateTimePickerCurrentDate.Value = dateTimePickerCurrentDate.Value.AddDays(-7);

DateToNumberWeek();

DateToWeek();

FillHeaders();

PrintSubjects();

}

private void dataGridViewTimeTable\_CellMouseDown(object sender, DataGridViewCellMouseEventArgs e)

{

if (e.RowIndex < 0 ||e.ColumnIndex < 2)

{

return;

}

dataGridViewTimeTable.Rows[iIndex].Cells[JIndex].Selected = false;

iIndex = e.RowIndex;

JIndex = e.ColumnIndex;

// this.Text = String.Format("строка{0}столбец{1}", iIndex, JIndex);

dataGridViewTimeTable.Rows[iIndex].Cells[JIndex].Selected = true;

if (e.Button == MouseButtons.Left)

{

int index = GetSubjectIndex();

if (index != -1)

{

textBoxSubject.Text = subjects[index].name;

textBoxTeacher.Text = subjects[index].teacher;

textBoxClassRomm.Text = subjects[index].ClassRoom;

richTextBoxHomeWork.Text = subjects[index].homeWork;

buttonEdit.Enabled = true;

}

else

{

textBoxSubject.Clear();

textBoxTeacher.Clear();

textBoxClassRomm.Clear();

richTextBoxHomeWork.Clear();

buttonEdit.Enabled = false;

}

}

}

private void добавитьToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (dataGridViewTimeTable.Rows[iIndex].Cells[JIndex].Value != null)

{

MessageBox.Show("Добавить сюда ничего нельзя");

}

else

{

checkBoxShowExInformation.Checked = true;

currentAction = Actions.add;

OnEdite();

}

}

private void buttonApply\_Click(object sender, EventArgs e)

{

if (textBoxSubject.Text == String.Empty || textBoxClassRomm.Text == String.Empty || textBoxTeacher.Text == String.Empty)

{

MessageBox.Show("Ошибка:Заполнены не все поля");

return;

}

if (currentAction == Actions.add)

{

subjects.Add(

new Subject()

{

name = textBoxSubject.Text,

teacher = textBoxTeacher.Text,

ClassRoom = textBoxClassRomm.Text,

homeWork = textBoxCurrentWeek.Text,

index = iIndex,

date = days[JIndex - 2]

}

);

textBoxSubject.Clear();

textBoxTeacher.Clear();

textBoxClassRomm.Clear();

richTextBoxHomeWork.Clear();

}

if (currentAction==Actions.edit)

{

int index = GetSubjectIndex();

subjects[index].name = textBoxSubject.Text;

subjects[index].teacher = textBoxTeacher.Text;

subjects[index].ClassRoom = textBoxClassRomm.Text;

subjects[index].homeWork = richTextBoxHomeWork.Text;

}

offEdit();

ClearTimeTable();

PrintSubjects();

SevesubjectsToFile();

currentAction = Actions.empty;

}

private void удалитьToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (dataGridViewTimeTable.Rows[iIndex].Cells[JIndex].Value == null)

{

MessageBox.Show("ОШИБКА ЯЧЕЙКА ПУСТА");

return;

}

else

{

int index=GetSubjectIndex();

subjects.RemoveAt(index);

ClearTimeTable();

PrintSubjects();

}

}

private void buttonEdit\_Click(object sender, EventArgs e)

{

currentAction = Actions.edit;

OnEdite();

}

private void textBoxClassRomm\_TextChanged(object sender, EventArgs e)

{

}

private void dataGridViewTimeTable\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void dateTimePickerCurrentDate\_ValueChanged(object sender, EventArgs e)

{

DateToNumberWeek();

DateToWeek();

FillHeaders();

}

}

}

## 1.4. Возможности приложения и практическое применение

Созданное мною приложение можно использовать на других компьютерах, ноутбуках. Для того чтобы его можно было использовать в других школах, необходимо сделать заполнение времени уроков самим пользователем, добавив в дополнительные возможности ещё одно поле. Можно также увеличить количество уроков в день. Данный органайзер может быть полезен, не только ученикам, но и учителям.

Аналогичное приложение может отображать занятость кабинетов, классов, учителей.

Есть - как и во всяком программном обеспечении - не только свои плюсы, но и минусы.

### Плюсы электронного ежедневника

1. **Всегда под рукой.** Современный электронный ежедневник – это программа для компьютера или приложение для телефона. То есть компьютерный или телефонный ежедневник не займет дополнительного места. Использовать первый вариант можно на рабочем месте, второй – более универсальный, он всегда рядом.
2. **Позволяет анализировать.** Электронный ежедневник помогает не только планировать школьную жизнь, а значит - добиваться успехов, но и в случае надобности, восстановить в памяти внесённые в органайзер события.
3. **Не оттягивает рук.** Пользуясь электронным ежедневником, можно носить с собой «архивы», которые не будут весить ни грамма;
4. **Исключает разбор почерка.** Не надо ломать голову над каракулями, как это случалось в эпоху бумажных органайзеров.
5. **Редактирует без грязи.** Можно легко удалять и редактировать информацию.
6. **Интуитивно понятен.** Интерфейс приближен к дневнику, знакомому обучающемуся с 1-го класса. Нет перегруза таблицами, столбцами/строками и окнами.
7. **Даёт ощущение стабильности.** Это, конечно, больше психологический аспект. И всё же, использование электронного органайзера позволяет не отвлекаться как в случае с новым ежедневником. В электронном продукте всегда всё предсказуемо и на местах. И так из года в год.

### Минусы электронного ежедневника

1. К минусам виртуальных ежедневников можно отнести его **зависимость от устройства**. Ежедневник для компьютера окажется недоступным в случае отключения электроэнергии.
2. Кроме того, владельцам электронных ежедневников стоит учитывать **возможность взлома или повреждения программы**, если ваш компьютер подхватил вирус.

# Заключение

# Мой органайзер был протестирован среди в параллели 5-х классов на уроке информатики. Результатом работы я доволен, поэтому такой электронный органайзер я хотел бы иметь на смартфоне, это удобнее, поскольку всегда с собой.

# Итак, я неплохо отработал навык работ в Visual Studio С#, справился с поставленными задачами. В будущем хочу научиться создавать приложения для смартфонов и других мобильных устройств. Важной задачей, для себя, считаю, изучение языков программирования и дальнейшую разработку программ, облегчающих обучение или организацию учебного процесса.

# Список Интернет-ресурсов:

1. <http://www.secreti.info/page19-vr.html>
2. <https://www.visualstudio.com>
3. <http://cpu.h17.ru/net/6/>
4. <http://softcreate.narod.ru/csharp.html>
5. <http://www.programmer-lib.ru/csharp.php>
6. <http://easyprog.ru/index.php?id=423&option=com_content&task=view>